L4: Entry 10 of 18 File: USPT Aug 31, 2004

DOCUMENT-IDENTIFIER: US 6785768 B2
TITLE: Computer system and process for transferring streams of data between multiple storage units and multiple applications in a scalable and reliable manner

Brief Summary Text (23):
In another aspect, data is recovered in a distributed data storage system having a plurality of storage units for storing the data, wherein segments of the data and redundancy information stored on the storage units are randomly distributed among the plurality of storage units, when failure of one of the storage units is detected. To recover the data, segments of which copies were stored on the failed storage unit are identified. The storage units on which the redundancy information corresponding to the identified segments was stored are identified. The redundancy information is used to reconstruct a copy of the identified segments, which are then randomly distributed among the plurality of storage units. Such data recovery may be used in combination with the read and write functionality of a file system or distributed storage system described herein.

Drawing Description Text (19):
FIG. 15 illustrates a structure of buffer memories for supporting playback of two streams of motion video data and four streams of associated audio data at a client;

Drawing Description Text (21):
FIG. 17 is a flowchart describing how a client requests a storage unit to transfer data from primary storage into a buffer in one embodiment;

Detailed Description Text (18):
Referring again to FIGS. 1A and 1B, when an application 44 requests access to a selected segment of data on one of the storage units 42, the storage unit places the request on a queue 48 that is maintained for the storage unit. Applications may make such requests independently of each other or any centralized control, which makes the system more readily scalable. Where the redundancy information is a copy of a segment, the selection of a storage unit to which a request is sent may be controlled such that random fluctuations in the load applied by multiple applications 44 on multiple storage units 42 are balanced statistically and more equally over all of the storage units 42. For example, each request from an application may be processed by the storage unit that has the shortest queue of requests. With any kind of redundancy information, the transfer of data between applications and storage units may be scheduled to reduce network congestion. The requests for data may be performed in two steps: a pre-read request which transfers the data from disk to a buffer on the storage unit, and a network transfer request which transfers data over the network from the buffer to the application. To process these two different requests, the queue 48 may include a disk queue and a network queue.

Detailed Description Text (22):
A segment of the data is created by the capturing system in step 121. The size of the segment may be, for example, one quarter, one half or one megabyte for motion video information. Audio information may be divided into, for example, segments having a size such as one-quarter megabyte. In order to provide alignment, if possible, of the segment size to divisions of storage and transmission, the size of the segment may be related, i.e., an integer multiple of, to an uncompressed or fixed data rate, disk block and track size, memory buffer size, and network packet (e.g., 64K) and/or cell sizes (e.g., 53 bytes for ATM). If the data is uncompressed or is compressed using fixed-rate compression, the segment may be divided at temporal sample boundaries which provides alignment between the image index and the segment table. Generally speaking, the segment size should be driven to be larger in order to reduce system overhead, which is increased by smaller segments. On the other hand, there is an

increased probability that a convoy effect could occur if the amount of data to be stored and segment size are such that the data is not distributed over all of the storage units. Additionally, there is an increased latency to complete both disk requests and network requests when the segment sizes are larger.

Detailed Description Text (33):
Referring again to FIG. 3, after two storage units are selected, the current segment then is sent to each of the selected storage units in step 124 for storage. These write requests may be asynchronous rather than sequential. The capture system then may wait for all storage units to acknowledge completion of the storage of the segment in the step 126. When data is stored in real time while being captured, the data transfer in step 124 may occur in two steps, similar to read operations discussed in more detail below. In particular, the client first may request a storage unit to prepare a free buffer for storing the data. The storage unit may reply with an estimated time for availability of the buffer. When that estimated time is reached, the capture system may request the storage unit to receive the data. The storage unit then may receive the data in its buffer, then transfer the data in its buffer to its storage medium and send an acknowledgment to the capture system.

Detailed Description Text (36):
FIG. 4 is a flowchart describing in more detail how a storage unit stores a segment of the captured data or redundancy information. The storage unit receives the segment of data from a capturing system in step 140 and stores the data in a buffer at the storage unit. Assuming the storage unit uses data files for storage, the storage unit opens a data file in step 142 and stores the data in the data file in step 144. The catalog manager may specify the location where the segment should be stored. The data may be appended to an existing data file or may be stored in a separate data file. As discussed above, the storage unit or the catalog manager may keep track of segments by using a unique identifier for each segment and by storing a table mapping the segment identifier to its location on the storage unit, in step 145. This table may implement the data file abstraction on the storage unit. When the storage unit actually writes data to its main storage may depend on other read and write requests pending for other applications. The management of these concurrent requests is addressed in more detail below. The file then may be closed in step 146. An acknowledgment may be sent to the capturing system in step 148.

Detailed Description Text (84):
The data structure described above and used to represent multimedia programs may use multiple types of data that are synchronized and displayed. The most common example is a television program or film production which includes motion video (often two or more streams or tracks) with associated audio (often four or more streams or tracks). As shown in FIG. 15, the client computer may have a corresponding set 290 of memory buffers 294 allocated in the main memory. Each buffer may be implemented as a "serializing" buffer. In other words, the client inserts data received from a storage unit into these independently accessible portions and reads from the set of buffers sequentially. Since requests may be sent to several storage units and data may be received at different times for the same stream, the buffers may not be filled in sequence when written, but are read out in sequence to be displayed. In FIG. 15, the filled in buffers indicate the presence of data in the buffer. Any empty buffer may be filled at any time as indicated at 293 and 295. However, each set of buffers has a current read location 291 from which data is read and which advances as time progress as indicated in 297. A subset 292, 296 of these buffers may be allocated to each stream of data.

Detailed Description Text (85):
Each buffer in the set of buffers has a size that corresponds to a fixed number of segments of data, where the segment size is the size of file segments stored on the storage units. There may be several, e.g., four, audio buffers per stream 292 of audio data, where each buffer may contain several, e.g., four, segments. Similarly, each video stream 296 may have several, e.g., four, buffers each of which contains several, e.g., four, segments. Each of the buffers may be divided into independently accessible portions 298 that correspond in size to the size of data packets for which transfer is scheduled over the network.

Detailed Description Text (86):
Because the video and audio data may be stored in different data files and may be combined arbitrarily, better performance may be obtained if requests for data for these different streams on the client side are managed efficiently. For example, the client application may

identify a stream for which data can be read, and then may determine an amount of data which should be read, if any. A process for performing this kind of management of read operations is shown in U.S. Pat. No. 5,045,940. In general, the client determines which stream has the least amount of data available for display. If there is a sufficient amount of buffer space in the set of buffers for that stream to efficiently read an amount of data, then that data is requested. It is generally efficient to read data when the available space in memory for the selected stream is large enough to hold one network transmission unit of data. When it is determined that data for a stream should be requested, each segment of the data is requested from a storage unit selected from those on which the segment is stored.

Detailed Description Text (88):
When the client requests a segment of data for a particular data stream, the client selects a storage unit, in step 272, for the segment that is requested. This selection, in one embodiment where the redundancy is provided by copying each segment, will be described in more detail below in connection with FIGS. 17 and 18. In general, the storage unit with the shortest queue 48 (FIG. 1) may be selected. The client then reads the data from the selected storage unit for the segment, in steps 274 through 278. Step 274 may be understood as a pre-read step in which the client sends a request to a storage unit to read desired data from nonvolatile storage into faster, typically volatile storage. The request to the storage unit may include an indication of how much time is required from the time the request is made until that requested data must be received at the client, i.e., a due time. After a pre-read request is accepted, the client waits in step 276. The request is placed in the storage unit's queue 48, and the due time may be used to prioritize requests as described below. Data is transferred from the storage unit in step 278 after data becomes available in a buffer at the storage unit. This step may involve scheduling of the network usage to transfer the data to maximize efficiency of network utilization. The received data is stored in the appropriate buffer at the client, and ultimately is processed and displayed in step 280. If the segment is lost at the storage unit, the redundancy information may be used to reconstruct the segment.

Detailed Description Text (89):
There are several ways to initiate the pre-read requests, including selection of a storage unit, in step 274 and the data transfer in step 278. For example, the MediaComposer authoring system from Avid Technology, Inc., of Tewksbury, Mass., allows a user to set either a number of clips or an amount of time as a look-ahead value, indicating how far ahead in a composition the application should initiate read requests for data. A program schedule for a television broadcast facility also may be used for this purpose. Such information may be used to initiate selection of a storage unit and pre-read requests. Such pre-reads may be performed even if buffer space is not available in buffers 290 (FIG. 15), as is shown in European patent application 0674414A2, published Sep. 9, 1995. The amount of available space in the buffers 290 (FIG. 15) may be used to initiate data transfers in step 278 (FIG. 16), or to initiate both pre-reads (step 274) and data transfers (step 278).

Detailed Description Text (90):
One process which enables a client to make an adequate estimate of which storage unit has the shortest queue of requests, without requiring an exhaustive search of all the available storage units, will now be described in connection with FIGS. 17 and 18. First, the client sends a request with a threshold E1 to a first storage unit in step 330. The threshold E1 is a value indicating an estimate of time by which the request should be serviced. This estimate may be expressed as a time value, a number of requests in the disk queue of the storage unit, such as four, or other measure. The meaning of this threshold is that the request should be accepted by the storage unit if the storage unit can service the request within the specified time limit, for example. The client receives a reply from the storage unit in step 332. The reply indicates whether the request was accepted and placed in the disk queue of the storage unit or whether the request was rejected as determined in step 334. If the request is accepted, the client is given an estimate of time at which the data will be available in a buffer at the storage unit in step 336. For example, if the data for the requested segment already is in a buffer, the storage unit indicates that the data is immediately available. The client then may wait until it is time to request transfer of the data (step 278 in FIG. 16) some time after the estimated time has passed. If the request is rejected, an estimate of the amount of time the storage unit actually is likely to take, such as the actual size in number of entries of the disk queue, is returned from the storage unit. This actual estimate is added to a value K to obtain a threshold E2 in step 340. The value K may be two, if representing a number of disk queue entries. Threshold E1 and value K may be user-definable. A request is sent to a second storage

unit in step 342 indicating the threshold E2. The client then receives a reply in step 344, similar to the reply received in step 332. If this reply indicates that the request was accepted, as determined in 346, the client has an estimate of time at which the data will be available at the second storage unit, as indicated in step 336 after which the client may wait to schedule the data transfer. Otherwise, an unconditional request, one with a large threshold, is sent to the first storage unit in step 348. An acknowledgment then is received in step 350 indicating the estimate of time at which the data will be available in a buffer at the storage unit, as indicated at step 336.

Detailed Description Text (91):
The storage unit, on the other hand, does not know whether it is the first or second storage unit selected by the client when it receives a request. Rather, the storage unit simply receives requests as indicated in step 360. The threshold indicated in the request is compared to the storage unit's own estimate of the time the client will need to wait in step 362, for example by comparing the size of the disk queue of the storage unit to the specified threshold. If the threshold in the request is greater than the estimate made by storage unit, the request is placed in the disk queue and an estimate of the time when the data will be available in a buffer at the storage unit is determined in step 364. This estimate may be determined, for example, based on disk access speed, disk queue length and possibly a running average of recent performance. An acknowledgment is sent to the client in step 336 including the estimated time of availability of the data in the buffer at the storage unit. Otherwise, a rejection is sent in step 368 indicating this estimate, such as the actual size of the disk queue.

Detailed Description Text (92):
The storage unit may keep track of which segments are in which buffers on the storage unit. Segment data may be read from the storage medium into any free buffer or into a buffer occupied by the least recently used segment. In this manner, data for a segment may be immediately available in a buffer if that segment is requested a second time.

Detailed Description Text (93):
As an alternative, a client may use another method to select a storage unit from which data will be retrieved, as discussed below. After sending the request, the client may receive an acknowledgment from the storage unit indicating that the request is in the disk queue at the storage unit. Instead of receiving an estimate of time at which the data will be available in a buffer at the storage unit, the client may wait until a ready signal is received indicating that the storage unit has read the requested data into a specified buffer memory at the storage unit. During this waiting period, the client may be performing other tasks, such as issuing requests for other data segments, displaying data or processing data. One problem with this alternative is that the client accepts an unsolicited message, i.e., the ready signal from the storage unit, in response to which the client changes context and processes the message. The client could be busy performing other operations. Although this process does provide a more accurate estimate of the time at which data is available in a buffer at the storage unit, the ability to change contexts and to process incoming messages quickly involves more complexity at the client.

Detailed Description Text (96):
FIG. 19 illustrates one embodiment of queue 48, utilizing a disk queue 300 and a network queue 320. The disk queue has four subqueues 302, 304, 306 and 308, one for each of the playback, capture, authoring and service and maintenance client programs, respectively. Similarly, the network queue 320 has four subqueues 322, 324, 326 and 328. Each queue includes one or more entries 310, each of which comprises a request field 312 indicating the client making the request and the requested operation, a priority field 314 indicating the priority of the request, and a buffer field 316 indicating the buffer associated with the request. The indication of the priority of the request may be a deadline, a time stamp, an indication of an amount of memory available at the client, or an indication of an amount of data currently available at the client. A priority scheduling mechanism at the storage unit would dictate the kind of priority stamp to be used.

Detailed Description Text (97):
The priority value may be generated in many ways. The priority value for an authoring or playback system is generally a measure of time by which the application must receive the requested data. For example, for a read operation, the application may report how much data (in milliseconds or frames or bytes) it has available to play before it runs out of data. The

priority indication for a capture system is generally a measure of time by which the client must transfer the data out of its buffers to the storage unit. For example, for a write operation, the application may report how much empty buffer space (in milliseconds, frames or bytes) it has available to fill before the buffer overflows. Using milliseconds as a unit of measure, the system may have an absolute time clock that could be used as the basis for ordering requests in the queue 48, and all applications and storage units may be synchronized to the absolute time clock. If such synchronization is not practical, the application may use a time that is relative to the application that indicates how much time from the time the request is made that may pass until the requested data should be received by the client. Assuming low communication latency, the storage unit may convert this relative time to an absolute time that is consistent with the storage unit.

Detailed Description Text (98):
The storage unit processes the requests in its disk queues 302-308 in their priority order, i.e, operating on the requests in the highest priority queue first, in order by their priority value, then the requests in successively lower priority queues. For each request, the storage unit transfers data between the disk and the buffer indicated by the request. For a read request, after the request is processed, the request is transferred from the disk queue to the network queue. For a write request, the request is removed from the disk queue after the write operation completes successfully.

Detailed Description Text (99):
In one embodiment to be described in more detail below, the storage unit uses the network queue to prioritize network transfers in the process of scheduling those transfers. In this embodiment, clients request transfer of data over the network. If a storage unit receives two such requests at about the same time, the storage unit processes the request that has a higher priority in its network queue. For a read request, after the request is processed, the request is removed from the network queue. For a write request, the request is transferred from the network queue to the disk queue, with a priority depending on the availability of free buffers, after the transfer completes successfully. If the time has passed for a request in the network queue to be processed, the request may be dropped indicating that the client is no longer operating or did not request the network transfer in time.

Detailed Description Text (103):
Referring now to FIG. 20, the client process for transferring data over the network will now be described. At any point in time during the playback of a composition, each buffer has a segment of data associated with it and a time by which the data must be available in the buffer for continuous playback. As is known in the art, the application associates each of the buffers with a segment during the playback process. As shown above in connection with FIGS. 17 and 18, each segment that a client has preread has an associated estimated time by which the data will be available at the storage unit. Accordingly, the client may order the buffers by their due time and whether the requested data is expected to be available in a buffer at the storage unit. This ordering may be used by the client to select a next buffer for which data will be transferred in step 500. The client requests a communication channel with the storage unit in step 502, specifying a waiting time E3. This value E3 may be short, e.g., 100 milliseconds, if the client does not need the data urgently and if the client may perform other operations more efficiently. This value E3 may be longer if the client needs the data urgently, for example, so that it does not run out of data for one of its buffers. In step 504, the client receives a reply from the storage unit. If the storage unit indicates that the request is rejected, as determined in step 506, a revised estimated time is received with the message in step 508. This revised estimated time may be used to update the buffer list in step 510 from which buffers are selected. Processing returns to step 500 to select another buffer. A buffer for which the segment is on the same storage unit as the previously selected segment probably should not be selected. If the storage unit otherwise accepts the request, the data ultimately is received in step 518.

Detailed Description Text (104):
The process from the point of view of the storage unit will now be described in connection with FIG. 21. The storage unit receives a request from a client in step 520 indicating waiting time E3. If the data is not yet available in the buffers at that storage unit, as determined in step 522, the storage unit rejects the request in step 524 and computes a revised estimated time which is sent to the client. If the data is otherwise available and the network connection of the storage unit is not busy, as determined in step 526, then the client becomes an "active

client" and the communication channel is granted by the storage unit in step 528, allowing data to be transferred. If the network connection of the storage unit is busy transferring data to another client, the storage unit maintains a request from a "waiting client," to which data is transferred after the data transfer for the "active client" is completed. In order to determine whether the current client should be the "waiting client," the storage unit estimates a time by which the transfer could occur, in step 530, based on the number of requests with earlier deadlines in the network queue multiplied by the network transmission time for each request. If the computed estimated time of availability is greater than the waiting time E3, indicating the client is not willing to wait that long, as determined in step 532, the request is rejected in step 524. Also, if the specified priority of this request is lower than the priority for any current waiting client, as determined in step 534, the request is rejected in step 524. Otherwise, the request from any current waiting client is rejected in step 536 and this new client is designated as the current waiting client. When a transfer to the active client is completed, the waiting client becomes the active client and the data is transferred.

Detailed Description Text (105):
In order to transfer data from a client to a storage unit, a similar process may be used for scheduling the network transfer and for transferring the data from a buffer in the storage unit to nonvolatile storage. From the point of view of the client, this process will now be described in connection with FIG. 22. This process may be used to implement step 124 and 126 in FIG. 3.

Detailed Description Text (106):
Unlike the process of reading in which the client may place data into an arbitrary point within its set of buffers, the data to be transferred to a storage unit typically comes from a read pointer from a set of buffers used by the capture system. The capture system typically produces one or more streams of video information as well as one or more streams of audio information. Accordingly, the capture system may select one of the data streams according to the amount of free buffer space in the stream to receive captured data. This buffer at the current read pointer of the selected stream is selected in step 600. A write request is then sent to the storage unit in step 602. The request includes an identifier for the segment, a due time or other priority value, and a threshold E4 indicating an amount of time the client is willing to wait. The due time is used by the storage unit to prioritize network transfer requests. The threshold E4 is used by the client, similar to threshold E3 discussed above, to permit the client to efficiently schedule its own operations. The client, after sending the request to the storage unit, eventually receives a reply in step 604. If the reply indicates that the write request was rejected, as determined in step 606, the reply includes an estimated time by which the storage unit will be available to receive the data. This estimated time may be used by the client to schedule other operations. If the storage unit accepts the request to write the data, the client then sends, in step 608, a portion of the segment of the data to the storage unit. A reply may be received in step 610 indicating whether or not the write request was successful, as analyzed in step 612. A failure may involve recovery processes in step 614. Otherwise the process is complete as indicated in step 616.

Detailed Description Text (107):
From the point of view of the storage unit, the storage unit receives the write request from the client in step 620. The request indicates a due time or other priority stamp which is used to place the request within the network queue. The storage unit then determines in step 622 if a buffer is available for receiving the data. The storage unit may make such a buffer available. In the unlikely event that no buffers are available, the request may be rejected in step 624. Otherwise, a request is put in the network queue in step 626 indicating the buffer allocated to receive the data, its priority stamp, and other information about the transfer. Next, the storage unit determines if the network connection is busy in step 628. If the network connection is not busy, the storage unit accepts the request in step 630 and sends a message to this effect to the client. The client then transfers the data which is received by the storage unit in step 632 and placed in the designated buffer. If the designated buffer is now full, as determined in step 634, the buffer is placed in the disk queue with an appropriate priority stamp in step 636. The storage unit's processing of its disk queue will eventually cause the data to be transferred from the buffer to permanent storage. Otherwise, the storage unit waits until the client sends enough data to fill the buffer as indicated in step 638.

Current US Cross Reference Classification (1):
707/205

CLAIMS:

1. A process for transferring temporal media data over a network from a plurality of storage units for playback on a client, wherein the temporal media data is divided into segments distributed over the plurality of storage units, wherein each storage unit has persistent storage and one or more buffers for storing data read from persistent storage for transmission over the network; wherein the client has a sequence of playback buffers, wherein each playback buffer temporarily stores a segment of the temporal media data before playback, and wherein data in the sequence of playback buffers is played back according to the sequence of the playback buffers, such that each playback buffer has a playback time, the process comprising: issuing requests to the storage units to read segments of the temporal media data from the persistent storage into the one or more buffers on the storage units; ordering empty playback buffers in the sequence of playback buffers by playback time and whether data for placement in each playback buffer is expected to be available in the one or more buffers in the storage units; selecting the playback buffer with the earliest playback time and for which the data for placement in the playback buffer is expected to be available in the one or more buffers in the storage units; and accessing the storage units for the data for placement in the selected playback buffer; repeating the steps of issuing, ordering, selecting and accessing during playback.

11. A client system for transferring temporal media data over a network from a plurality of storage units for playback on the client system, wherein the temporal media data is divided into segments distributed over the plurality of storage units, wherein each storage unit has persistent storage and one or more buffers for storing data read from persistent storage for transmission over the network, wherein the client system comprises: a sequence of playback buffers, wherein each playback buffer temporarily stores a segment of the temporal media data before playback, and wherein data in the sequence of playback buffers is played back according to the sequence of the playback buffers, such that each playback buffer has a playback time, means for issuing requests to the storage units to read segments of the temporal media data from the persistent storage into the one or more buffers on the storage units; means for ordering empty playback buffers in the sequence of playback buffers by playback time and whether data for placement in each playback buffer is expected to be available in the one or more buffers in the storage units; means for selecting the playback buffer with the earliest playback time and for which the data for placement in the playback buffer is expected to be available in the one or more buffers in the storage units; means for accessing the storage units for the data for placement in the selected playback buffer; and means for controlling the means for issuing, ordering, selecting and accessing during playback.

Previous Doc Next Doc Go to Doc#